Types of Relationships in Object Oriented Programming (OOP)

- Inheritance "is a" relationship
- Association "has-a" relationship
- Composition "part-of" relationship
- Aggregation "has-a" relationship

Types of Relationships in Object Oriented Programming (OOP)

In other words, the object of one class may use:
Services (Data Members, Methods)
provided by the object of another class. This kind of relationship is termed as **association**

Types of Relationships in Object Oriented Programming (OOP)

الإتصال علاقة منفصل

Association: is a connection or relation between two separate classes that are set up through their objects.

من خلال

أحادي ثنائي

Association: represents a general unary or binary relationship that describes an activity between two classes.

نشاط

Types of Relationships in Object Oriented Programming (OOP)

Unified Modeling Language (UML)

is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems

Types of Relationships in Object Oriented Programming (OOP)

List of UML Diagram Types

Structure Diagrams:

- Class Diagram
- Component Diagram
- Deployment Diagram
- Object Diagram
- Package Diagram
- Profile Diagram
- Composite Structure Diagram

Behavioral Diagrams:

- Use Case Diagram
- Activity Diagram
- State Machine Diagram
- Sequence Diagram
- Communication Diagram
- Interaction Overview Diagram
- Timing Diagram

UML tools for code generation?

Enterprise Architect from Sparx.

..

..

..

..

Types of Relationships in Object Oriented Programming (OOP)

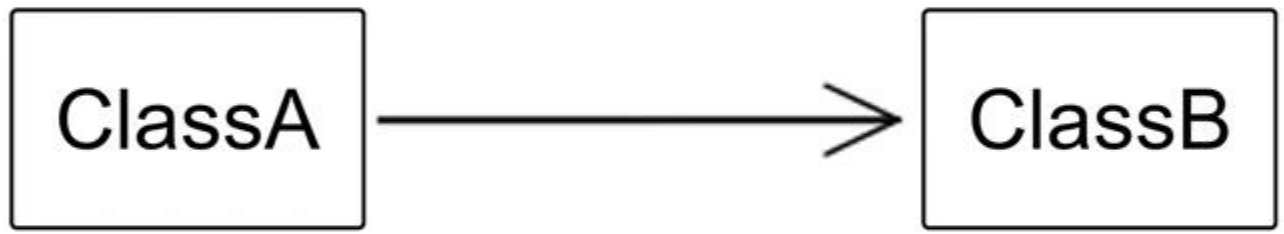
Association

Unary

Binary

Types of Relationships in Object Oriented Programming (OOP)

Association (Unary)

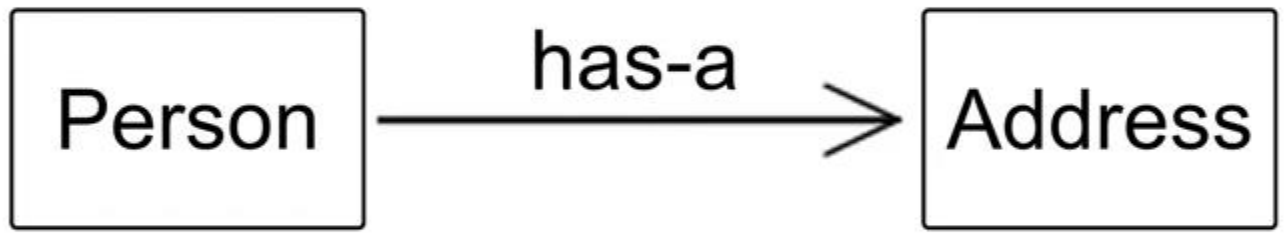


ClassA knows about ClassB

ClassB knows nothing about ClassA

Types of Relationships in Object Oriented Programming (OOP)

Association (Unary)



Person knows about Address

Address knows nothing about Person

Types of Relationships in Object Oriented Programming (OOP)

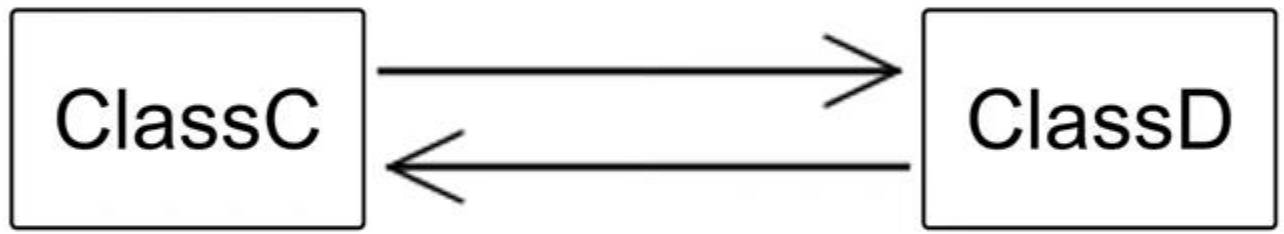
Association (Unary)

Person knows about Address

Address knows nothing about Person

Types of Relationships in Object Oriented Programming (OOP)

Association (Binary)

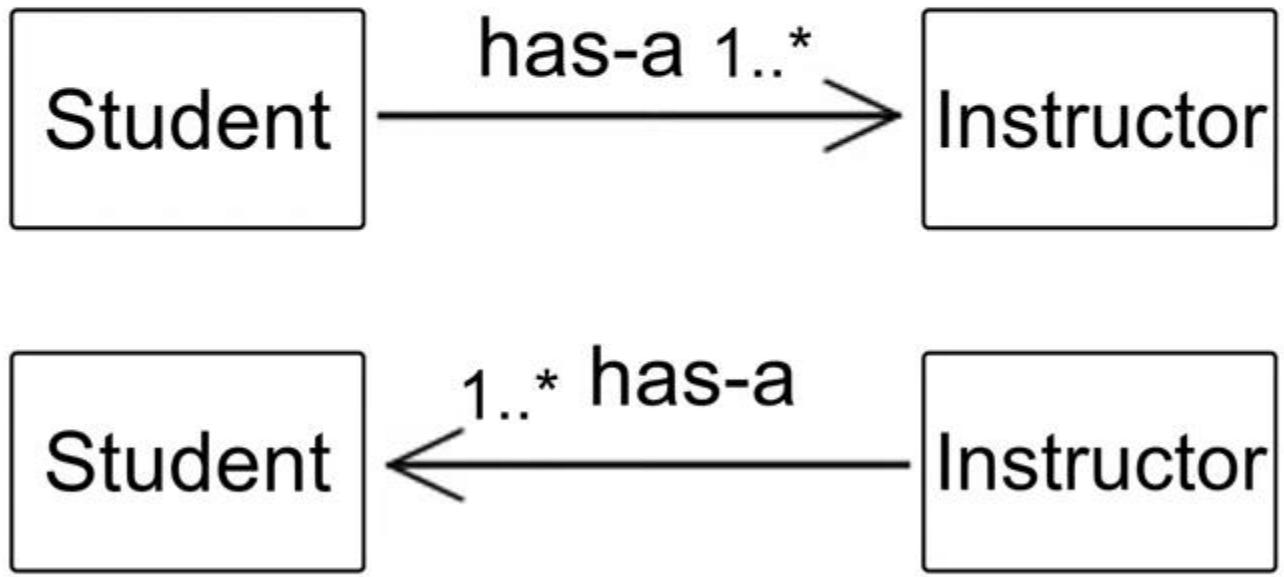


ClassC knows about ClassD

ClassD knows about ClassC

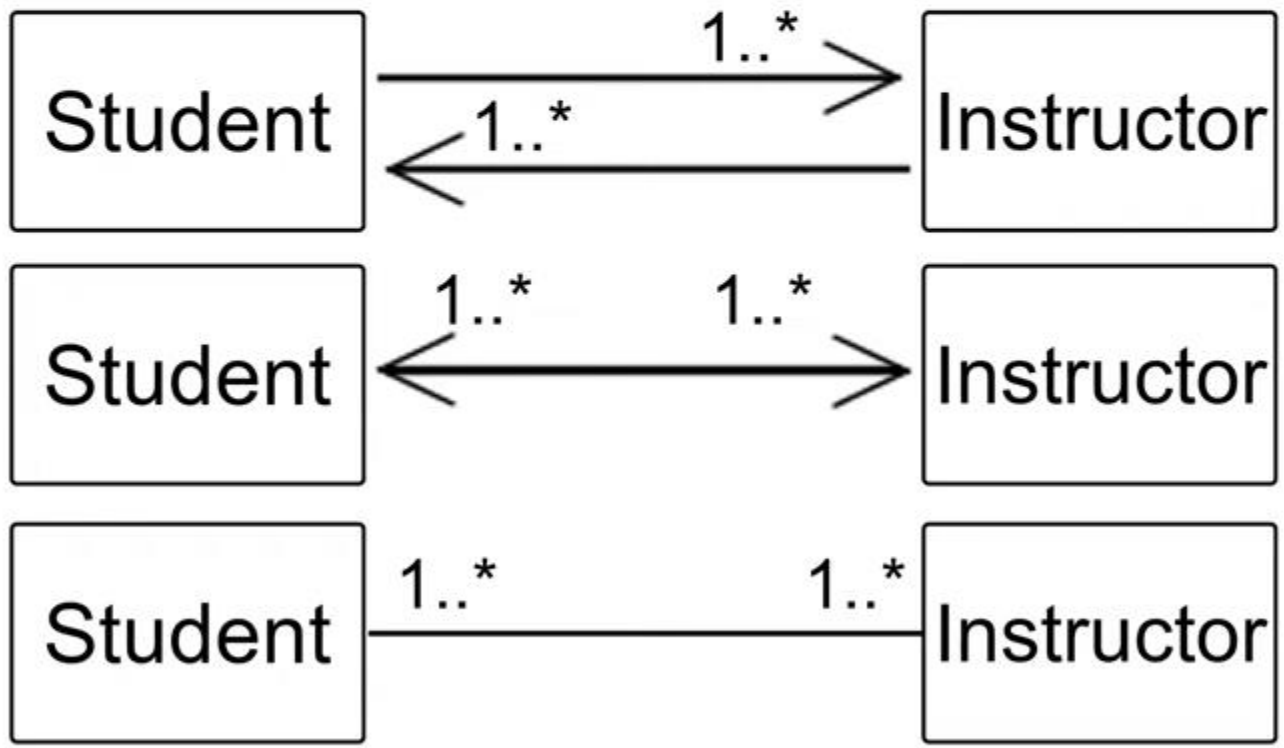
Types of Relationships in Object Oriented Programming (OOP)

Association (Binary)



Types of Relationships in Object Oriented Programming (OOP)

Association (Binary)



Types of Relationships in Object Oriented Programming (OOP)

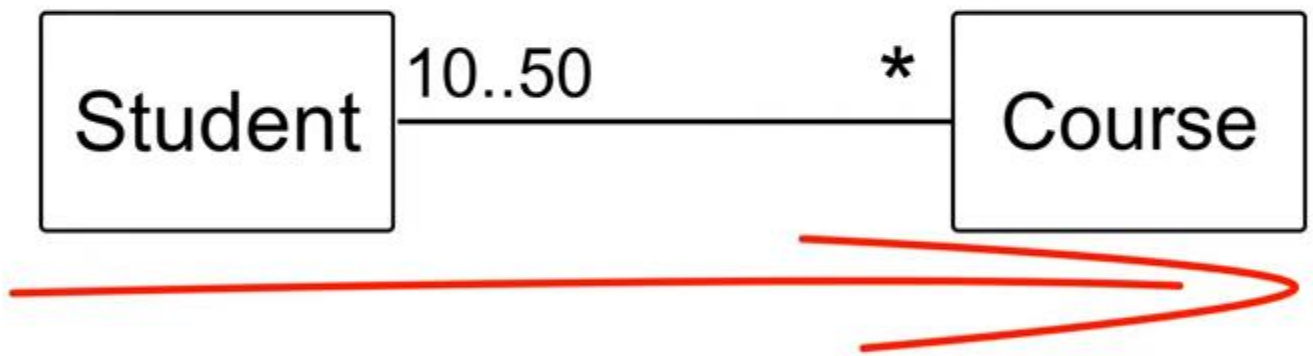
Association (Binary)

Multiplicity: تعددية

0..1	an optional instance (zero or one)
n	exactly n instances
*	zero or more instances
1..*	one or more instances
n..m	n to m instances

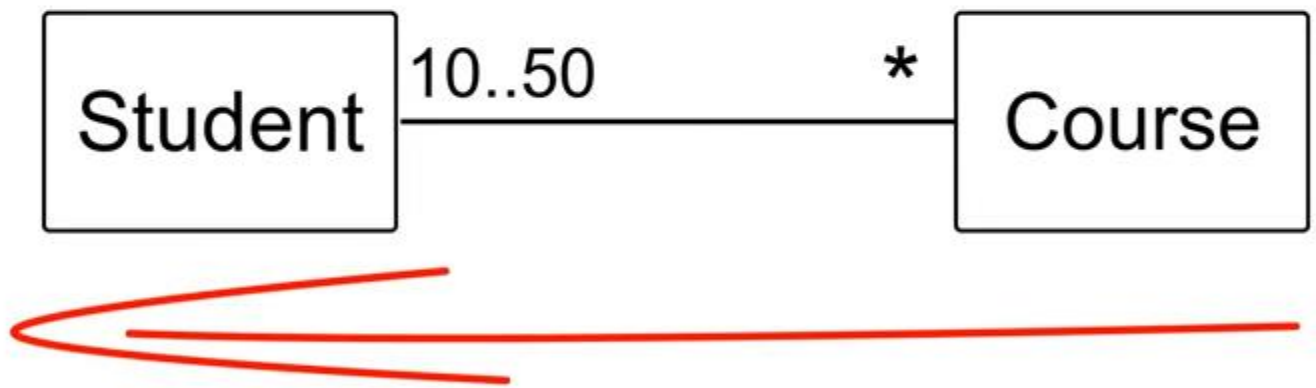
Types of Relationships in Object Oriented Programming (OOP)

Association (Binary)



Types of Relationships in Object Oriented Programming (OOP)

Association (Binary)



Types of Relationships in Object Oriented Programming (OOP)

3 types of Unary Associations:

- Association most general

- Aggregation "has-a" relationship
- Composition "part-of" relationship

whole-part relationships

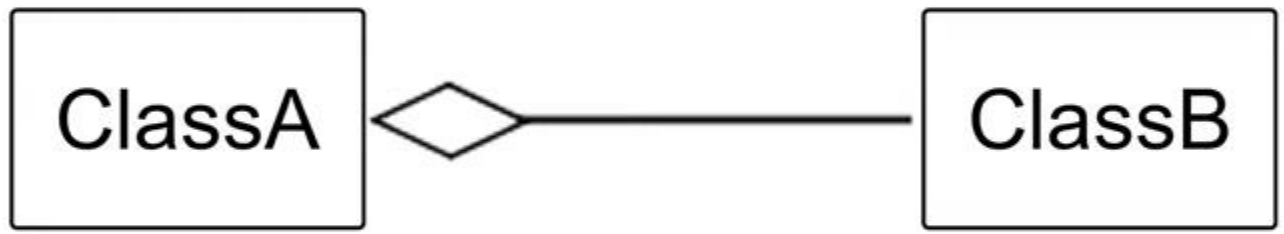
Types of Relationships in Object Oriented Programming (OOP)

Aggregation "has-a" relationship

Types of Relationships in Object Oriented Programming (OOP)

Aggregation "has-a" relationship

Aggregation weak relationship



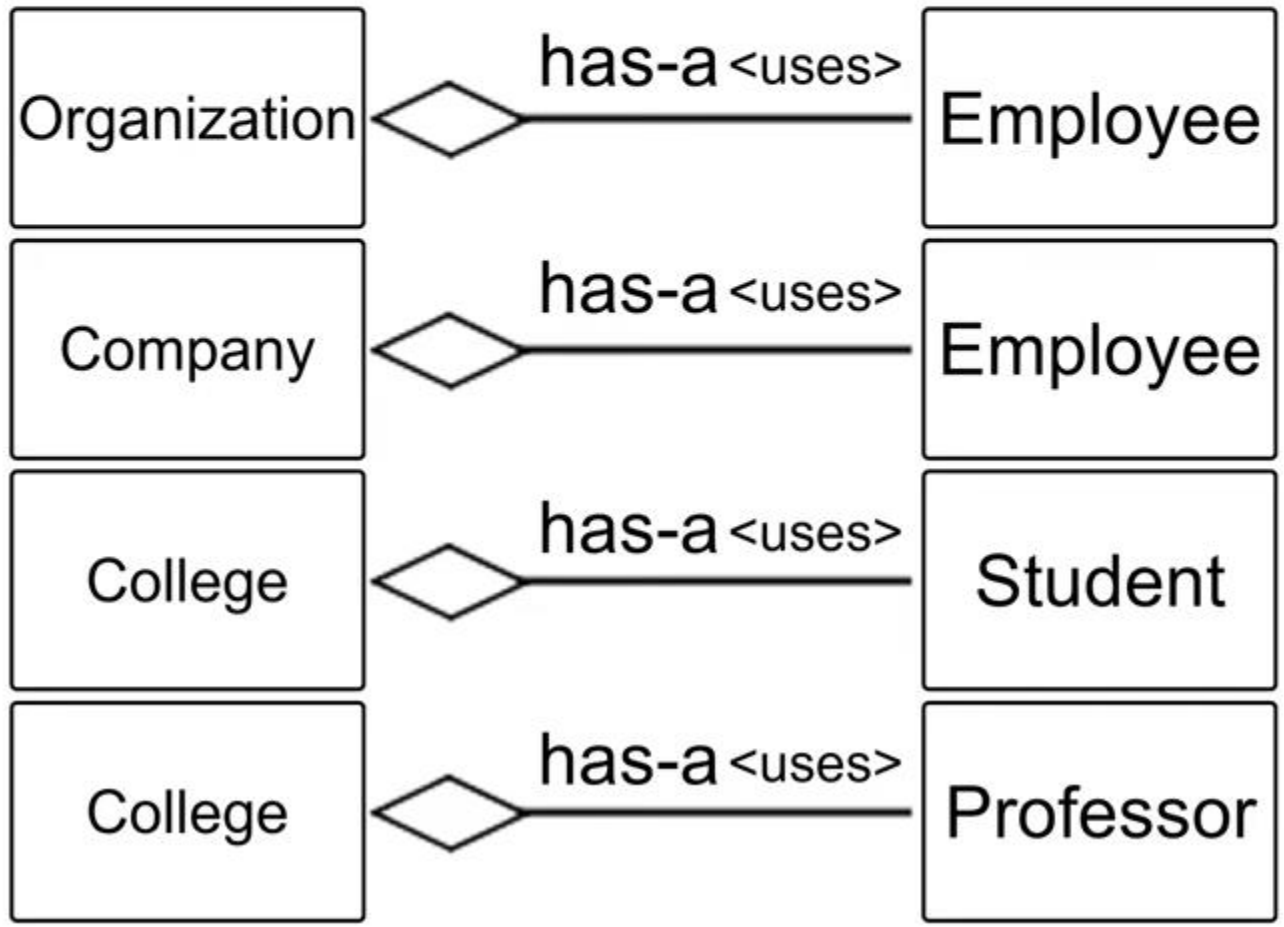
ClassB is part of ClassA

A "uses" B = Aggregation : B exists independently (conceptually) from A

Types of Relationships in Object Oriented Programming (OOP)

Aggregation "has-a" relationship

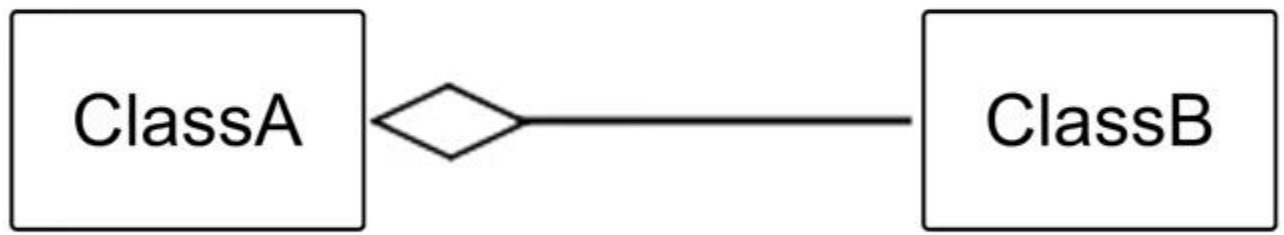
Aggregation weak relationship



Types of Relationships in Object Oriented Programming (OOP)

Aggregation "has-a" relationship

Aggregation weak relationship

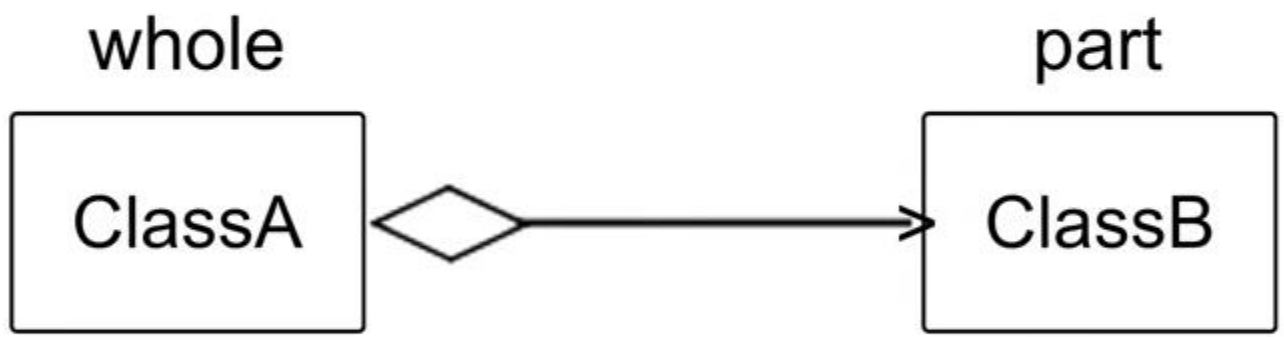


If aggregation (ClassA) is deleted, The parts (ClassB) associated with it are not deleted.

Types of Relationships in Object Oriented Programming (OOP)

Aggregation "has-a" relationship

Aggregation weak relationship



Aggregation is a special form of association. It is a relationship between two classes like association, however it's a **directional association**, which means it is strictly a **one-way association**. It represents a **HAS-A relationship**.

Types of Relationships in Object Oriented Programming (OOP)

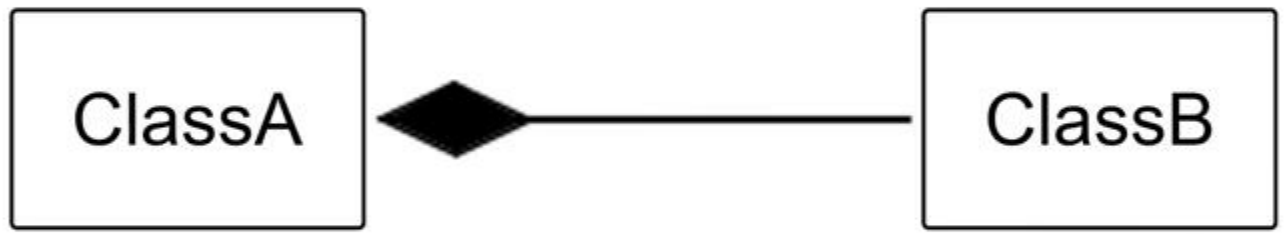
3 types of Unary Associations:

- Association most general
- Aggregation "has-a" relationship
- Composition "part-of" relationship

whole-part relationships

Types of Relationships in Object Oriented Programming (OOP)

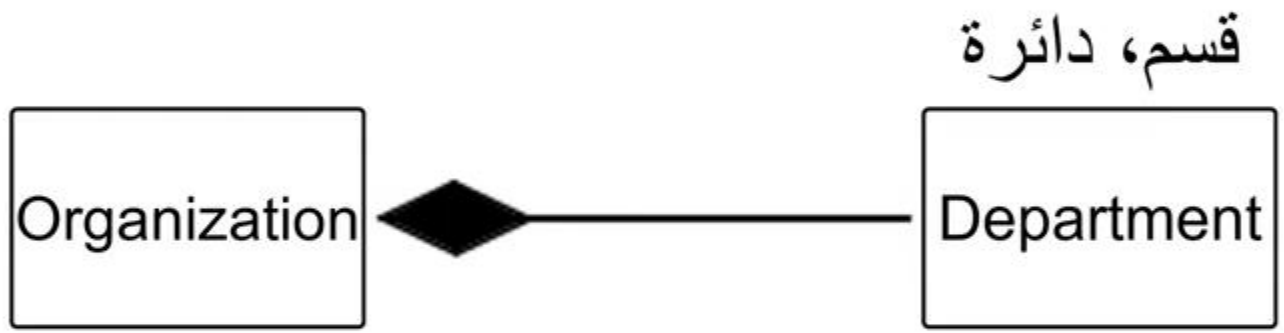
Composition "part-of" relationship



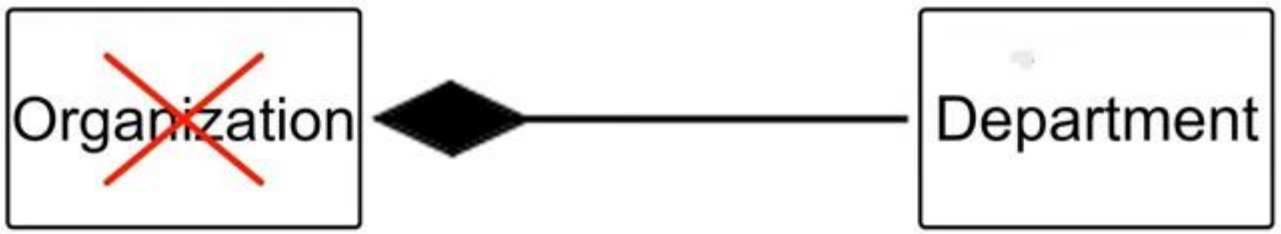
ClassB has **no meaning** or **purpose** in the system without **ClassA**

Types of Relationships in Object Oriented Programming (OOP)

Composition "part-of" relationship

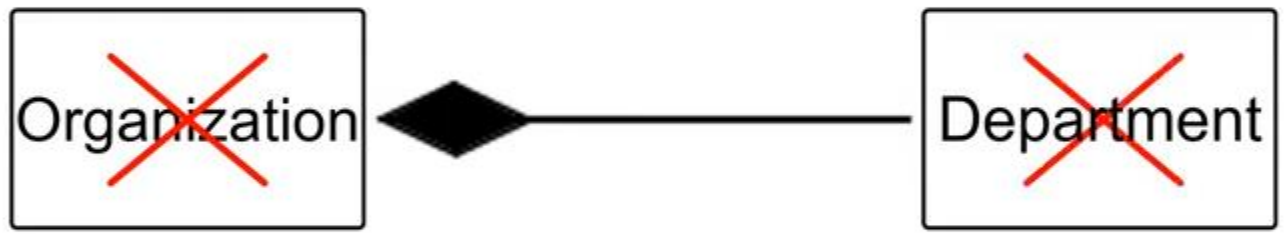


Composition "part-of" relationship



Types of Relationships in Object Oriented Programming (OOP)

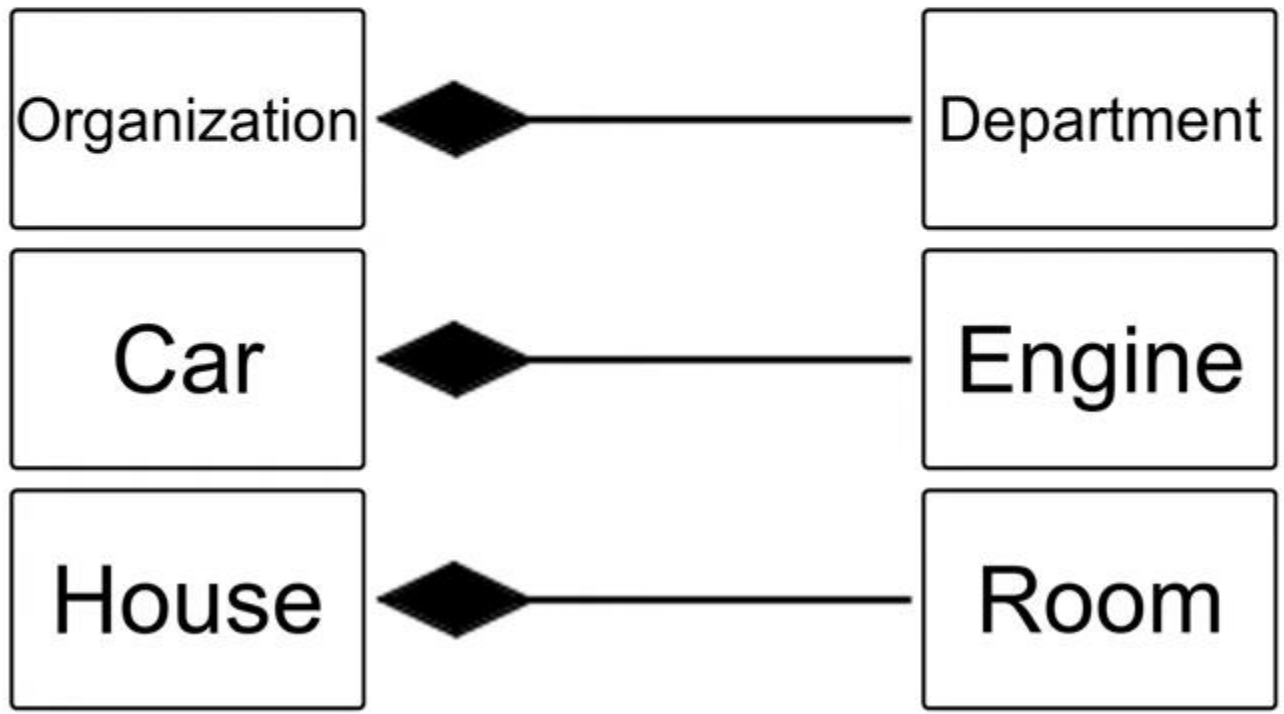
Composition "part-of" relationship



Types of Relationships in Object Oriented Programming (OOP)

Composition "part-of" relationship

Composition: strong relationship

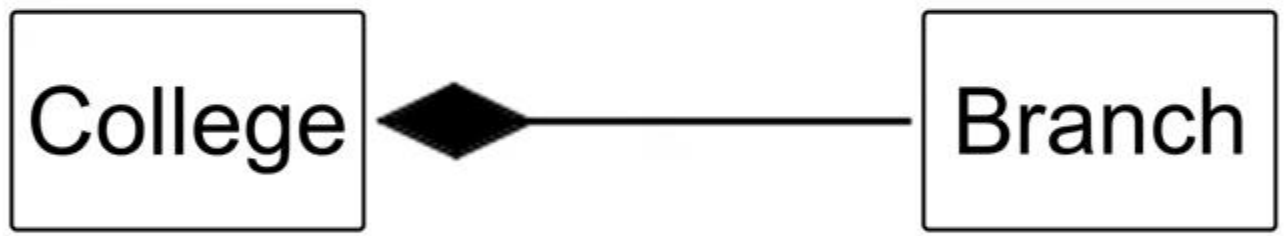


Types of Relationships in Object Oriented Programming (OOP)

Composition "part-of" relationship

Composition: strong relationship

whole **part-of**

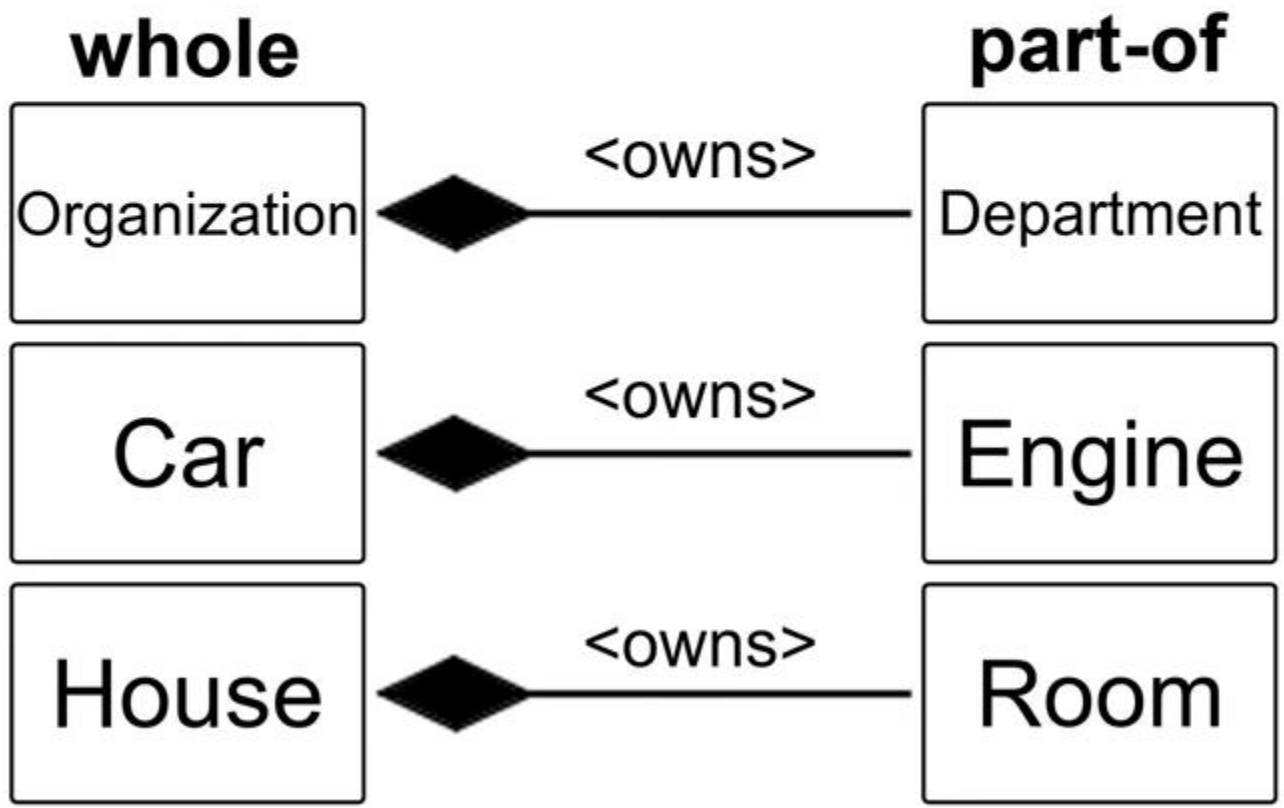


(CIS, CS, MIS, SE)

Types of Relationships in Object Oriented Programming (OOP)

Composition "part-of" relationship

Composition: strong relationship



Types of Relationships in Object Oriented Programming (OOP)

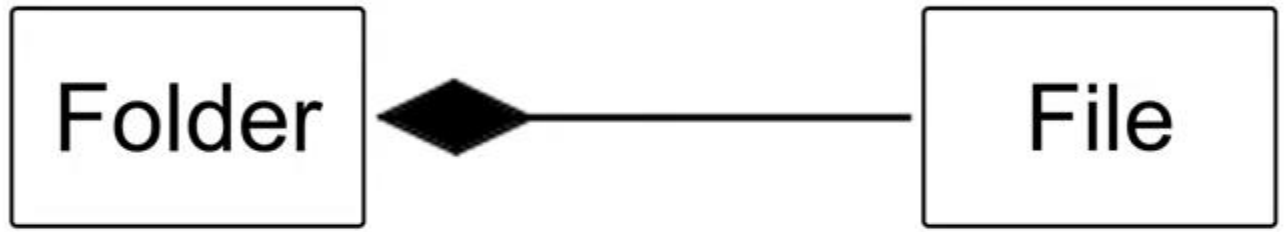
ObjectA "uses" ObjectB = Aggregation : ObjectB exists independently (conceptually) from ObjectA

Aggregation (If you remove "whole", "Part" can exist – "No Ownership")

Composition is a specialized form of Aggregation. It is also called "death" relationship.

Types of Relationships in Object Oriented Programming (OOP)

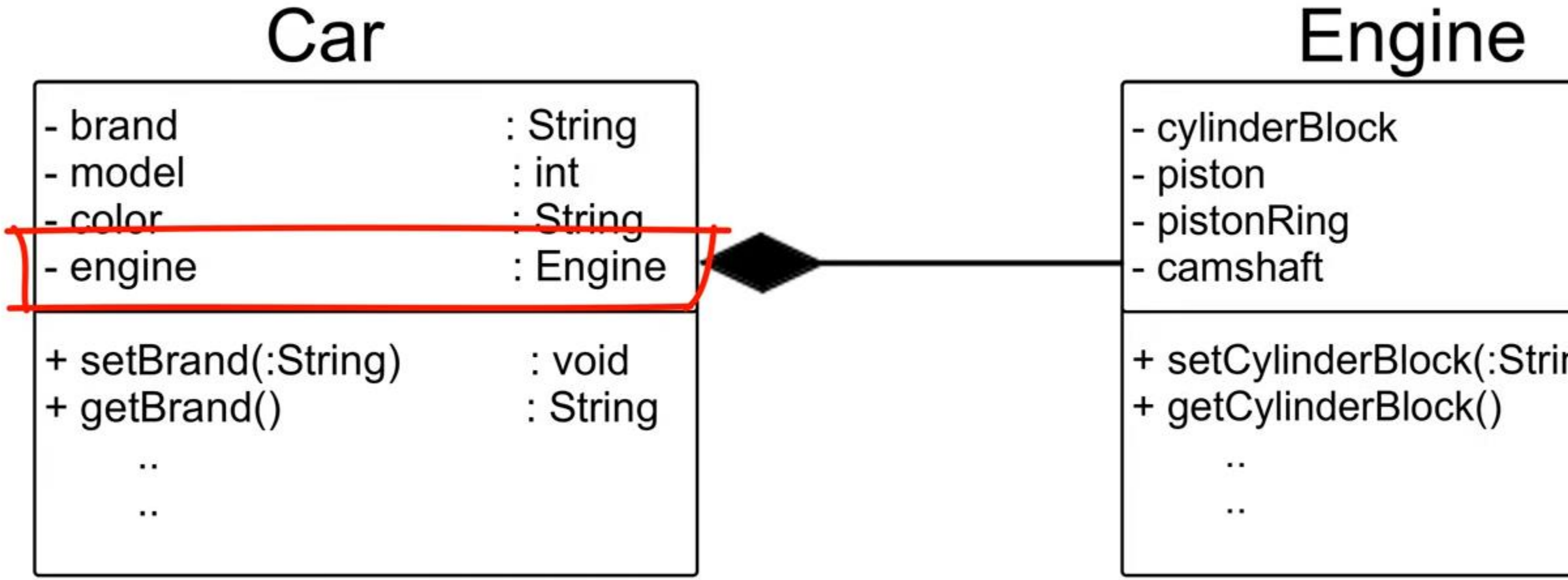
Composition



Types of Relationships in Object Oriented Programming (OOP)

In both **aggregation** and **composition** object of one class "owns" object of another class.

Composition



Types of Relationships in Object Oriented Programming (OOP)

Association

```
public class A {
    void testMethod(B objectB) {
    ...
    }
};
```

Types of Relationships in Object Oriented Programming (OOP)

Composition

```
public class A {
    private B objectB = new B();
    void testMethod() {
    ...
    }
};
```

Types of Relationships in Object Oriented Programming (OOP)

Composition

```
public class A {
    private B objectB = new B();
    void testMethod() {
    ...
    }
}
```

Aggregation

```
public class A {
    private B objectB;
    A(B objectB) {
        this.objectB = objectB;
    }
}
```

Types of Relationships in Object Oriented Programming (OOP)

Composition

```
public class A {
    private B objectB = new B();
    void testMethod() {
    ...
    }
}
```

Aggregation

```
public class A {
    private B objectB;
    A(B objectB) {
        this.objectB = objectB;
    }
}
```

```
public class A {
    private B objectB;
    A(B objectB) {
        this.objectB = objectB;
    }
}
```

```
public static void main (String[] args){
        B objectB = new B();
        A obj = new A(objectB);
        ..
        ..
        ..
        ..
}
```

```
public class A {
    private B objectB;
    A(B objectB) {
        this.objectB = objectB;
    }
}
```

```
public static void main (String[] args){
    B objectB = new B();
    A obj = new A(objectB);
    ..
    ..
    ..
    ..
}
```

if we remove objectA, objectB doesn’t affect